Resistant Automotive Miniature Network

Camille Gay
Toyota Motor Corporation
Tokyo, Japan
camille.gay @toyota-tokyo.tech

Tsuyoshi Toyama
Toyota Motor Corporation
Tokyo, Japan
tsu-toyama @toyota-tokyo.tech

Hisashi Oguma
Toyota Motor Corporation
Tokyo, Japan
oguma @toyota-tokyo.tech

Abstract

Automotive testbeds are important in enabling a high level of security in modern cars, but currently available solutions are expensive and not optimized for evaluating physical attacks. As a result, security researchers cannot freely study attacks on automotive networks and their countermeasures and cannot easily perform potentially destructive tests or evaluate the impact of hardware manufacturing tolerances. In addition, expensive testbeds often need to be shared between researchers, preventing them from customizing their testbed and bringing it to a home office. To address this issue, we developed a relatively inexpensive open-source automotive testbed called the Resistant Automotive Miniature Network (RAMN), which fits in a printed circuit board with the size of a credit card. The testbed consists of four electronic control units (ECUs) connected to a common controller area network (CAN bus) compatible with flexible data-rate (CAN-FD). It can operate under the same environmental conditions in which actual ECUs operate. It is small enough to fit into equipment used for automotive testing and was designed to interface with popular tools used by hardware security researchers. It can be connected in a closed loop with the self-driving simulator CARLA to emulate a functional automotive network. By releasing this testbed, we aim to offer more freedom to security researchers. We also hope it will be another step to make the automotive hardware and software industry more open and ultimately enable better security in cars.

I. Introduction

Automotive security is a field of study that is recently attracting considerable attention from the information security community. To demonstrate attacks and countermeasures, researchers have used either real cars or testbeds that emulate a car's architecture. Security conferences often feature a "car hacking village" where one can find many homemade automotive testbeds that are built with parts stripped from old cars [1] [2]. Designing an automotive security testbed is not a trivial task, as designers must consider many parameters, such as cost, size, usability, reproducibility, fidelity to reality, and non-disclosure agreement (NDA) requirements. As a result, these testbeds usually have very restricted use cases that they try to achieve well. Hobbyists' testbeds aim to help in education and vulnerability finding. Academic testbeds function to enable a proper and reproducible evaluation of security technologies, while guaranteeing a safe environment. However, none of the currently available testbeds are optimized for physical security testing. They do not fit in the testing equipment used by hardware security researchers [3]. In addition, because they do not use automotive-grade components, they would break if exposed to the same conditions that real electronic control units (ECUs) operate in (e.g., high temperature). Another issue is that testbeds are expensive, so researchers are often constrained to share a single testbed, preventing them from experimenting freely because they cannot afford to break it or permanently modify it. This factor also prevents researchers from evaluating many physical attacks and associated countermeasures, potentially leaving the next generation of cars at risk of having vulnerabilities. Finally, working from home is becoming the norm for many researchers in 2020, but expensive testbeds might be too risky to be kept at home.

To address these issues, we developed the Resistant Automotive Miniature Network (RAMN), an open-source testbed optimized for physical testing. It is contained within a printed circuit board (PCB) with the size of a credit card, so it can fit in automotive testing equipment and equipment used by hardware security researchers. It mostly embarks automotive-grade components that can resist temperatures up to 150 °C and can therefore operate in the same conditions as real ECUs. It is designed to be inexpensive, so that researchers can own many testbeds and not have to worry about breaking or monopolizing them.

The remainder of the paper is organized as follows: In Section 2, we provide a quick introduction to what "automotive grade" means and how it impacts the security level of cars. In Section 3, we present related works, derive the requirements for our ideal testbed, and then describe our design. In Section 4, we evaluate the testbed. In Section 5, we discuss the testbed's limitations. In Section 6, we briefly conclude the paper.

II. Background and Related Works

A. *What "automotive grade" means*

A modern car's architecture relies on several ECUs with different purposes. For instance, the airbag ECU is in charge of detecting shocks and triggering the airbag. Typically, ECUs can only use hardware and software that are qualified for automotive use.

Avoiding failures that could lead to catastrophic consequences is the highest priority. However, at the software level, bugs are inevitable. At the hardware level, failures of individual components will always occur at a certain rate. The goal of ISO 26262 [4] is to ensure that the probability of a catastrophic event to happen because of bugs and component failures is negligible. This field of study is known as functional safety. Specifically, ISO 26262 defines criticality levels called the Automotive Safety Integrity Level (ASIL). The "QM" level is assigned to non-critical ECUs, the ASIL A level is assigned to ECUs of low criticality, and the ASIL D level is assigned to ECUs whose failure would endanger people's life. To be used in an ECU, the hardware and software must prove that they comply with requirements associated with their respective ASIL levels.

In addition to the safety requirements defined by ISO 26262, there are also reliability requirements that components must satisfy. Concretely, "automotive-grade" hardware reliability requirements are defined by the Automotive Electronics Council (AEC). The AEC released several documents describing tests that components for automotive use must pass. Most notably, AEC-Q100 [5] describes tests for integrated circuits, and AEC-Q200 [6] describes tests for passive components. AEC-Q100 has four grades (0 to 3), where grade 0 has the harshest requirements, with an operation temperature of up to 150 ℃. Tests include temperature cycling (e.g., 2000 times alternating from −55 °C to 150 °C), high-temperature storage (e.g., 175 ℃ for 1000 hours), and high-temperature operation (e.g., 150 ℃ for 1000 hours). They also include electromagnetic compatibility tests defined by SAE J1752/3. To pass these tests, components might need to be designed with older field-proven technologies and conservative design rules, different from the design rules of general-purpose components. Finally, in addition to industry standards' requirements, manufacturers add their own requirements based on their experience and constraints. For example, they might only allow the use of components with external pins to facilitate visual inspection after manufacturing.

Microcontrollers are often the main processing unit of an ECU. Considering the requirements stated above, microcontroller manufacturers usually offer a special line of products dedicated to automotive use. For example, Renesas offers the RH850 family of microcontrollers [7], and Infineon offers the AURIX family [8]. These microcontrollers are typically more expensive and require signing an NDA to obtain datasheets and user guides. They usually have special safety features [9]; for example, critical ECUs' microcontrollers are likely to have two central processing units (CPUs) executing the same code in a lock-step configuration to reduce the probability of not detecting a CPU hardware failure.

The software toolchain that generates the code running on microcontrollers must also comply with special requirements. These include compilers, runtime libraries, real-time operating systems (RTOSs), frameworks, and applications. Automotive software development requirements are defined by ISO26262 [4], ISO/IEC 15504 [10] (Automotive SPICE), and the Motor Industry Software Reliability Association (MISRA) [11]. Software vendors typically have a special line of products for automotive use: Green Hills Compilers [12] or Wind River Diab Compiler [13] are examples of automotive-grade compilers. The EB tresos [14] and ETAS RTA-OS [15] are examples of automotive-grade RTOS. AUTOSAR [16] and its predecessor OSEK [17] are examples of automotive-grade frameworks. It should be noted that some ECUs, such as telematics ECU and infotainment units, might be exempted from many software and hardware requirements because they are less safety critical and are located in the car's interior – the less demanding environment of a car.

Car manufacturers (referred to as original equipment manufacturer (OEMs)) do not often design and test ECUs themselves – they outsource this task to the so-called "Tier-1" manufacturers. Tier-1 manufacturers typically develop ECUs following processes defined by ISO/TS 16949 [18] and ISO26262 [4] and processes specific to each OEM. To be approved for use in a car, an ECU must pass a series of tests:

- Reliability testing (e.g., high-temperature operation, temperature shocks, high humidity, overvoltage, cranking, electrostatic discharge).
- Electromagnetic compatibility testing, including EMI testing (limiting noise coming out of the ECU) and EMS testing (resisting the noise coming in the ECU).
- Mechanical testing (e.g., acceleration, shocks, vibrations).
- Foolproof testing (e.g., operator dropping the ECU, battery plugged with wrong polarity, battery jumpstarted with a wrong donor connection).
- Others.

Different OEMs will require different tests with different passing thresholds, but they will typically not vary significantly. It is therefore not uncommon to see the same ECU being reused across different OEMs. ECUs are designed using both guidelines mandated by OEMs and guidelines from the Tier-1 manufacturers themselves. These guidelines include the following:

- Regular electronic design guidelines, such as keeping bypass capacitors close to a component's power supply pins and preventing unwanted antennas.
- Automotive-specific guidelines, such as ensuring a sufficient gap between high-voltage traces to prevent electromigration and doubling the number of vias.
- Design for manufacturability (DFM) guidelines, to ensure that the ECU is suitable for mass production.

B. How automotive grade impacts security

Considering the requirements and processes stated above, ECUs are different in terms of hardware and software from consumer electronics, such as smartphones [19]. Smartphones have a restricted temperature range (0 °C–85 °C), limited operating life (2–3 years), and a remarkably high failure rate (300 parts per million). In comparison, ECUs in the engine compartment need to resist to a temperature range of −40 °C to 150 °C, with an operating life of more than 10 years and a failure rate close to zero. In addition, while consumer electronics can feature anti-tampering and obfuscation techniques in their products, the automotive industry needs to always perform a failure analysis to quickly identify the cause of malfunctions and the risk that it happens again. If a smartphone abruptly failed, it would likely lead to no harm, the manufacturer would replace the defective smartphone, and the user would forget about it quickly. However, a failing airbag ECU would probably cause harm and require a speedy investigation that may result in a product recall. These constraints prevent the use of many anti-reverse engineering techniques, such as permanently locking debug ports and the use of encryption for some data. As a result, while smartphones use the latest technologies, ECUs use only well-proven technologies and are often tagged as the automotive industry "lagging behind."

One can wonder how the safety and reliability requirements of automotive-grade electronics impact the security of ECUs. The differences between MISRA-C and CERT-C have already been studied [20], and researchers have already shown that ECUs can be susceptible to software and hardware attacks [21] [22]. Safety features found in automotive microcontrollers, such as error-correcting code memory (ECC memory), make attacks significantly harder but not impossible [23]. Similarly, features meant to improve reliability and electromagnetic compatibility, such as RAM scrambling, have been shown to make attacks harder but not impossible [24]. In this context, we think that the security of automotive-grade electronics should be studied further.

C. Issues with currently available testbeds

Testbeds are extremely useful for security research, and many researchers have developed and shared the design of their own security testbeds. These designs range from software-only testbeds [25] to high-end automobile simulators [26]. Many hobbyists have built their own automotive testbeds made with parts stripped from old cars [1] [2]. OEMs also develop their own testbed: Toyota Motor Corporation proposed a portable, adaptable testbed named PASTA [27], which is similar to hobbyists' homemade testbeds but is made of reproducible open-source technologies.

Available security testbeds can fall into one of two categories: either they use automotive-grade technologies, or they do not. Automotive-grade testbeds have the advantage of being close to a real automotive environment, ensuring that the results would also be valid for a real car. However, the use of automotive technologies requires researchers to sign NDAs, preventing them from publishing their results. As such, these testbeds are often only used privately by major industry players. Testbeds built by hobbyists from old cars do not require signing an NDA, but they are made with black-box components, which cannot be easily customized. Oftentimes, researchers resort to a collection of Arduino and Raspberry Pi boards for their prototypes, which are not easily reproducible. Testbeds that are built using non-automotive technologies have a great advantage: they are less expensive, and documentation/software can be downloaded online. However, because they do not use automotive-grade technologies, researchers from the automotive industry that are skeptical of the results of a novel research paper based on a non-automotive testbed can always argue "but this would not happen on a real car with automotive technologies, because…" and ask for the results to be proven again on an automotive-grade testbed. Ideally, automotive microcontroller manufacturers and software vendors would open the specifications of their technologies to encourage research, but this is too unrealistic to expect as of 2020. We therefore need a solution that guarantees that testbeds are affordable and open, while not being too far from an equivalent automotive grade solution.

Currently available testbeds also have the problem of being expensive. This issue prevents many talented researchers from buying or borrowing one testbed to work from home. It also prevents even well-funded researchers from conducting potentially destructive research or applying permanent changes to their testbeds because they do not want to risk breaking their testbeds or they need to share the testbeds with other researchers whose research is incompatible.

D. Difficulties with physical testing

We decided to focus our research on "non-automotive grade" testbeds because testbeds requiring an NDA are too restrictive. One issue with open-source testbeds is that the usage of non-automotive-grade hardware makes them less suitable for physical testing because they use different technologies at the hardware level. Another issue with these testbeds is that they are often too inconvenient to bring into testing equipment. At best, they are the size of a suitcase. They are meant to be used at ambient temperature on a desk. However, real ECUs do not operate at ambient temperature on a desk and operate in extreme conditions, for example, 105 °C in the engine compartment, potentially during a storm. One cannot bring a suitcase in a clean room and put it in an autoclave or a scanning electron microscope. Even if they fit, the testbed would likely break before reaching 85 °C. However, in some microcontrollers, glitches are more likely to occur at high temperatures [28] [29] (with "high temperature" above 60 °C). Contrary to smartcards that can decide to shut down if a high temperature is reached, ECUs need to operate in extreme conditions and are therefore more exposed to attacks due to environmental stress. Therefore, automotive technologies need to be tested under various environmental conditions, not just at room temperature.

Researchers have shown that the aging of a device actually improves its resistance to static power analysis attacks [30] and Template attacks [31]. Aging the testbed on purpose enables researchers to ensure that a countermeasure that is valid on the first day of the car will still be valid after the car has aged. However, currently available automotive testbeds are destroyed by automotive accelerating aging processes. The high cost of testbeds also prevents researchers from evaluating technologies on several testbeds. However, real ECUs use components that have manufacturing tolerances (e.g., voltage, impedance). To demonstrate that a technology is suitable for automotive use, it should work on a variety of testbeds with small variations due to manufacturing tolerances and not just when it has been fine-tuned for one particular instance.

III. Design of an automotive-grade testbed

We demonstrate that the physical testing of automotive-grade electronics is important, but currently available testbeds are not suitable. To address this problem, we developed an open-source automotive testbed, which is a hybrid of automotive-grade hardware and non-automotive grade software, which is optimized for physical testing (e.g., side-channel analysis, fault injection, destructive testing). In this section, we first review physical security platforms in non-automotive sectors, extract preferable requirements, and then describe our design.

A. *Physical security testing platforms*

Many platforms have been developed to ease the evaluation of physical security attacks. The SASEBO project [32] offers a testbed optimized for testing cryptographic modules. FOBOS [33] and SCARF [34] have also been proposed as platforms to evaluate physical security countermeasures. The ChipWhisperer project [35], a popular side-channel and fault injection platform, also offers a "UFO Target" platform [36] for "attacking all sorts of embedded targets." Available physical testing platforms have the following similarities:

- They are contained within one PCB or within several connected PCBs.
- They feature low-noise power sources.
- They offer easy access to critical signals, such as clocks and power lines.
- They embed common tools, such as amplifiers and data acquisition tools.

While all these platforms are popular and well-suited for many cases, they are not made with automotive-grade technologies and are therefore not a solution to our problem. In addition, they focus on attacking one component (e.g., a microcontroller or cryptographic module) rather than a network of components (e.g., a controller area network (CAN)/CAN-flexible data-rate (CAN-FD) network).

B. *Testbed requirements*

According to our review of currently available platforms, we concluded that the requirements for an automotive security testbed optimized for physical testing should be

- **(R1)** The testbed should be compatible with currently available tools and platforms: there are many platforms and tools already available and maintaining the compatibility with available testbeds ensures researchers the freedom to switch between platforms.
- **(R2)** The testbed should be made of automotive-grade components: automotive-grade components have different characteristics.
- **(R3)** The testbed should be open source: This characteristic ensures that the results can be correctly analyzed and reproduced. It also ensures that the testbed can be improved or repurposed by other researchers.
- **(R4)** The testbed should be inexpensive: This characteristic enables researchers to own more than one testbed, so they can perform potentially destructive testing on them or apply nonreversible modifications. In 2020, this characteristic has also allowed researchers to work from home.
- **(R5)** The testbed should be small enough to fit in common testing equipment (e.g., autoclave, scanning electron microscope (SEM)). This characteristic enables researchers to perform their research using automotive testing equipment, for example, to simulate a high temperature/humidity environment. It also enables researchers to use hardware security testing equipment (e.g., a microprobing station).
- **(R6)** The testbed should be optimized for physical security testing (low-noise power sources, easy access to critical signals, and embedding common tools). This ensures that physical security attacks can be evaluated with high quality.

C. *Proposal*

We designed a CAN/CAN-FD automotive security testbed based on a single PCB, with the following characteristics:

- Using the same high-level architecture as PASTA [27] (**R1**)
- Optimized for mass production at a low cost (**R4**)
- Has the size of a credit card (**R4, R5**)
- Optimized for physical testing by featuring low-noise power sources and probes to access critical signals (**R6**)
- Using mostly AEC-Q100 [5] and AEC-Q200 [6] components resisting temperatures up to 150 °C (**R2**)
- Open source (**R3**)
- Standalone (USB powered and requires no equipment, such as a CAN adapter or programmer) (**R4, R6**)

D. *Detailed design*

1) *Reusing the PASTA architecture*

The testbed consists of four ECUs with the same architecture as PASTA [27]:

- Gateway ECU
- Powertrain ECU
- Chassis ECU

- Body ECU

These ECUs are connected to a common CAN/CAN-FD bus. The gateway ECU is additionally connected to a USB port and can be used as a general-purpose CAN/CAN-FD adapter using either the “slcan” protocol [37] or socketCAN drivers [38]. It can be reprogrammed over a USB using STM32’s built-in DFU bootloader [39]. The gateway ECU can also control the power supply unit of each ECU individually. By turning on/off the power supply of each ECU, the gateway ECU can program other ECUs over CAN using STM32’s built-in CAN bootloader. Therefore, all ECUs are independently reprogrammable over USB. The advantage of reusing the architecture of PASTA is that researchers who work with PASTA can immediately apply their results to RAMN. Another advantage is that RAMN can serve as an alternative to PASTA for researchers who cannot afford one. While PASTA only uses CAN bus technology, RAMN is compatible with both the CAN technology and its more recent evolution is CAN-FD. The block diagram of RAMN is shown in Figure 1. A picture of the board is shown on Figure 2.

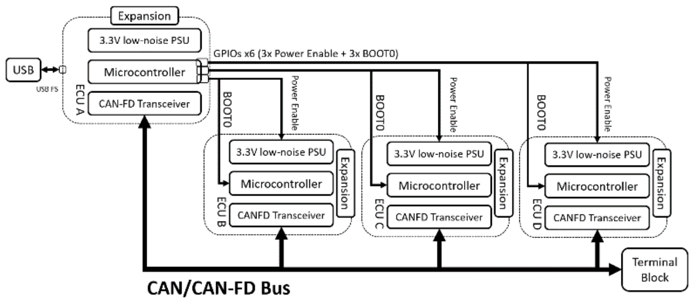


Figure 1 Block diagram of RAMN

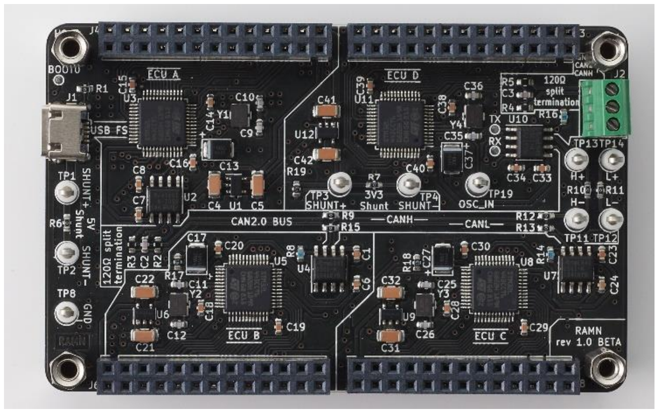


Figure 2 Picture of RAMN main board.

2) Mass producible at a low cost

To ensure that the PCB can be mass-produced inexpensively, it is designed using permissive design rules (e.g., a 0.15 mm track clearance) and with only two layers. In addition, the board uses only components with external pins (no QFN and no BGA packages) that are large enough for hand soldering (size 0608 or above). This means that the board can be easily reworked or entirely soldered by hand. As a result, RAMN can be produced and assembled by most PCB fabrication manufacturers within the low-price range. The board is also accessible to hobbyists and students who want to fabricate and assemble it themselves. There are many advantages associated with a low-cost testbed:

- It ensures that destructive tests (e.g., testing at high temperatures) can be performed with a reasonable budget.
- It ensures that researchers can have their own testbed instead of needing to share one. They can therefore customize it (e.g., reprogram the firmware and replace components) and modify physical characteristics (e.g., aging components) without impacting the work of other researchers. They can work from home.
- It enables researchers to verify their results on several testbeds, which will have slightly different characteristics due to the manufacturing tolerances of components. Doing so ensures that their results are valid not only for a particular hardware instance but for a greater manufacturing range. These factors are illustrated in Figure 3.

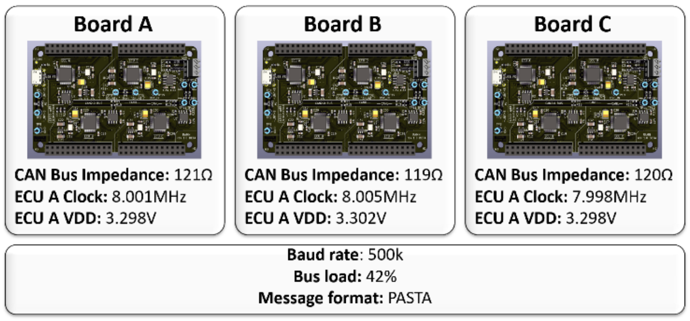


Figure 3 Effects of manufacturing tolerances on the testbeds’ physical characteristics.

3) Small form factor

The whole testbed fits into an 85.60 mm x 53.98 mm PCB (size of a credit card) and features four M3 holes to enable easy mounting on testing equipment.

Because of its small size, the testbed can fit in the following:

- Testing equipment of security researchers: Hardware security researchers often use special equipment [3] in clean rooms, such as microprobing stations, laser cutters, focused ion beam workstations, and SEM workstations.
- AEC-Q100 testing equipment: The testbed should fit into the same equipment used to qualify components for AEC-Q100 [5], such as HAST chambers and TEM cells.

4) Optimized for physical testing

Low cost and small form factor are meant to enable destructive and extreme condition testing. The most anticipated use cases of such a testbed are side-channel analysis and fault injection (glitching). Most of the time, these attacks will require access to critical signals, and the results will depend on the quality of the signals on the board. To ensure clean signals, the board features an individual low-noise power supply for each microcontroller, and the CAN/CAN-FD bus line is designed with 120 Ω differential impedance microstrip lines. The CAN/CAN-FD bus has split terminations at each end. To enable current monitoring and glitching of the power line, the board also features shunt resistors on the 3.3 V lines. To ensure easy instrumentation, the board features test probes on critical signals: clock, power, CAN RX, CANH, and CANL. The board has also been designed to be connected directly to the popular physical security evaluation framework "ChipWhisperer" [35].

5) *Automotive-grade components*

Because the goal is to enable researchers to use the testbed in extreme conditions, we selected components that conform to AEC-Q100 and AEC-Q200 grade 0 (150 °C temperature limit). Hence, the testbed should have the same physical characteristics as an actual automotive ECU. There are only two components that do not meet the automotive requirements: the USB connector and microcontrollers. Although there are automotive-grade USB connectors, they are uncommon enough that most people would not recognize them as a USB port at first glance, and they are less readily available. Instead, we tentatively selected a consumer's electronics connector. As for the microcontrollers, automotive-grade microcontrollers and their software toolchains are associated with high costs and restrictive NDAs. Instead, we selected microcontrollers with comparable features and properties. We initially selected STM32L443CC, which is a low-power ARM microcontroller with an operating temperature range of −40 °C to 125 °C (the same range as many automotive-grade microcontrollers for use outside of the engine compartment). It also features a CAN controller, an Advanced Encryption Standard (AES) engine, and ECC capability. Although not automotive grade, it is still close enough to an automotive microcontroller. For countries with restrictions on encryption engines, the board can also be assembled with STM32L433CC microcontrollers, which do not feature the AES engine. The board can also be populated by the more recent STM32L5 series, which features a CAN-FD controller and additional security capabilities, such as a TrustZone execution environment and a true random number generator (TRNG). The board is compatible with STM32L552 (version without AES engine) and STM32L562 (version with AES engine).

6) *Open source*

The choice of an STM32 microcontroller is also strongly motivated by the fact that the STM32 family has gained popularity within the maker community because of numerous evaluation boards (STM32 Nucleo boards [40]) and easy integration with popular open-source projects (e.g., FreeRTOS [41] and mbed TLS). To ensure that RAMN can easily be customized and reprogrammed, the software is built using the default RTOS supported by STMicroelectronics: FreeRTOS. Because we make the project open source, researchers are free to remove the RTOS or replace it with another one and customize it the way they need.

7) *Standalone*

Currently available testbeds require external equipment, such as external CAN/CAN-FD adapters, to observe signals [27]. With RAMN, one of the ECUs can be programmed as a CAN/CAN-FD adapter. The advantage of this research is that there is no need to bring a CAN/CAN-FD adapter and programmer in the testing environment. There is also less signal distortion introduced by external components and less risk that the CAN/CAN-FD adapter is destroyed when testing under extreme operating conditions.

In contrast to PASTA, RAMN does not embed sensors and actuators. Instead, it can be fitted with "expansion headers" like those that can be found in Arduino and Raspberry Pi boards. Researchers can design their own expansion boards and connect them using one of the many interfaces accessible (e.g., SPI, I2C, Universal Asynchronous Receiver Transmitter (UART), analog to digital converter (ADC), digital to analog converter (DAC)). We designed expansions that are stackable and compatible with each other; that is, they can be used at the same time. We designed the expansions for external memories (e.g., electrically erasable programmable read-only memory (EEPROM), static random access memory (SRAM), ferroelectric random access memory (FRAM)), screens (several models from Adafruit [42]), Trusted Platform Module (TPM), connection to ChipWhisperer, debug connections (JTAG and probes), and sensors and actuators (e.g., dashboard LEDs, brake/accelerator/steering potentiometers)

An example of a RAMN setup with several expansion boards is illustrated in Figure 4.

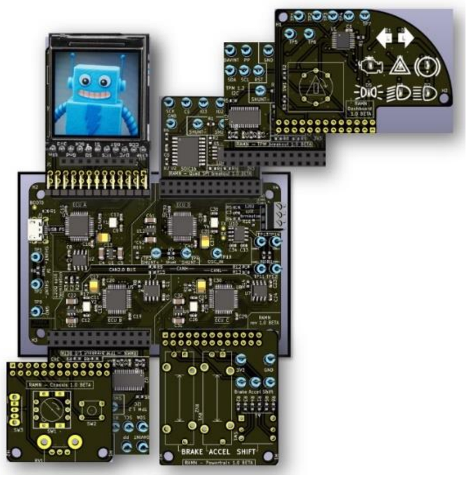


Figure 4 Breakout of an example setup of RAMN board expanded with TPM, external memories, screens, and sensors/actuators.

IV. EVALUATION

We fabricated boards using common processes and tolerances. We programmed their software using STM32CubeIDE environment and FreeRTOS [41]. Figure 5 shows a simple RAMN setup with sensors and actuators, and Figure 6 shows a RAMN setup with many expansion boards (TPM, memory, and debugger). All functions worked as intended: the four ECUs can be reprogrammed over USB, the CAN/CAN-FD bus is fully functioning, and CAN/CAN-FD frames can be observed over USB with slcan or socketCAN.

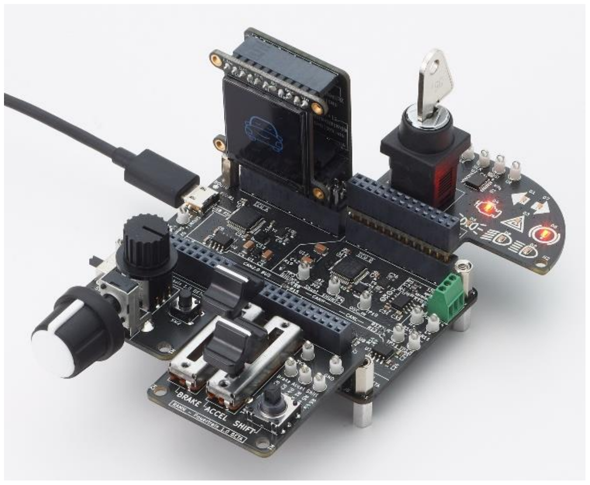


Figure 5 Picture of a RAMN setup with sensors/actuators.

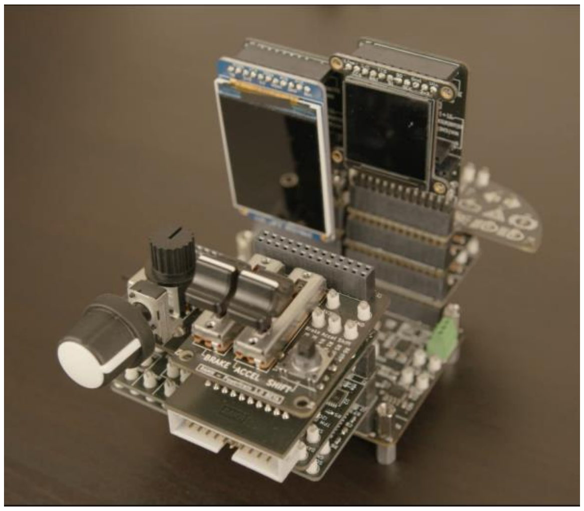
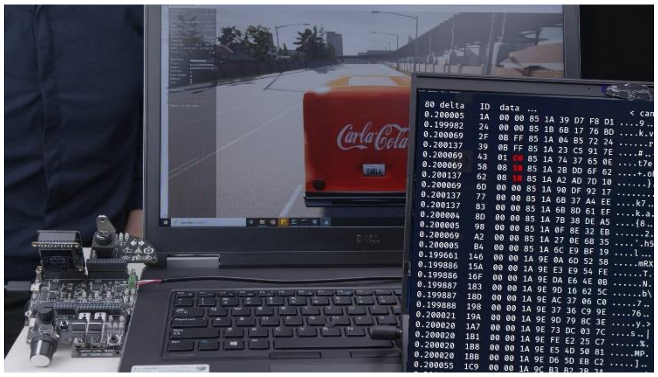


Figure 6 Picture of a RAMN setup with several expansions (TPM, external memories, debugger).

A. Integration with the self-driving simulator CARLA

As a default environment, we programmed the board to be used in conjunction with the popular driving simulator CARLA [43], as shown in Figure 7. We modified CARLA so that RAMN is integrated in a closed loop with the simulator; that is, all commands (e.g., brake, steering) must first be processed by the powertrain, body and chassis ECUs before they are passed back to CARLA's real-world simulation. For example, the self-driving algorithm can send a brake request on CAN, which the powertrain ECU will receive, process, and then send a brake command. The gateway ECU will pass back that command to CARLA's real-world simulation, which will trigger the brakes in the virtual world. In parallel, the body ECU will also receive the brake command and activate the stop lamp LED. With this closed-loop integration, virtual cars can be controlled by the sensors on RAMN, and values existing only in the virtual world (e.g., car speed) can also be found on the CAN/CAN-FD bus and trigger actuators (e.g., LEDs). We verified that by using the CAN IDs and formats defined by PASTA [44], the car would be comfortably controllable using the potentiometers on the sensor boards. The delay added by the introduction of the closed-loop control was not significant enough to prevent the default self-driving algorithm of CARLA to function correctly. Despite a heavy bus load of approximately 42%, no CAN bus message drop or significant delay was observed.



```
80 delta    ID  data ...                  < can
0.200005    1A  00 00 85 1A 39 D7 F8 D1 ....9..
0.199982    24  00 00 85 1B 6B 17 76 BD ....k.v
0.200069    2F  0B FF 85 1A 04 B5 72 24 ......r
0.200137    39  0B FF 85 1A 23 C5 91 7E ....#..
0.200069    43  01 C0 85 1A 74 37 65 0E ....t7e
0.200069    58  08 10 85 1A 2B DD 6F 62 ....+.o
0.200137    62  08 10 85 1A A2 AD 7D 10 ......}.
0.200069    6D  00 00 85 1A 90 DF 92 17 ........
0.200137    77  00 00 85 1A 6B 37 A4 EE ....k7..
0.200137    83  00 00 85 1A 6B 8D 61 EF ....k.a.
0.200004    8D  00 00 85 1A 7B 38 DE A5 ....{8..
0.200005    98  00 00 85 1A 0F 8E 32 EB ......2.
0.200069    A2  00 00 85 1A 27 0E 68 35 ....'.h5
0.200005    B4  00 00 85 1A 6C E9 BF 19 ....l...
0.199661   146  00 00 1A 9E 0A 6D 52 58 .....mRX
0.199886   15A  00 00 1A 9E E3 E9 54 FE ......T.
0.199886   16F  00 00 1A 9E DA E6 4E 0B ......N.
0.199887   183  00 00 1A 9E 9D 16 62 5C ......b\
0.199887   18D  00 00 1A 9E AC 37 06 C0 .....7..
0.199888   198  00 00 1A 9E 37 36 C9 9E ....76..
0.200021   19A  00 00 1A 9E 9D 79 8C 3E .....y.>
0.200020   1A7  00 00 1A 9E 73 DC 03 7C ....s..|
0.200020   1B1  00 00 1A 9E FE E2 25 C7 ......%.
0.200020   1B8  00 00 1A 9E E5 4D 50 81 .....MP.
0.200020   1BB  00 00 1A 9E D6 5D EB C2 .....]..
0.200055   1C9  00 00 1A 9C B3 B2 2B 7A
```

Figure 7 Picture of a RAMN setup used in a closed loop with the CARLA simulator, connected to the CAN bus visualization tool candump (Linux).

B. Evaluation of physical attacks

To ensure that the board can be used to evaluate physical attacks, we used ChipWhisperer Pro [35] to perform basic analyses, as shown in Figure 8.

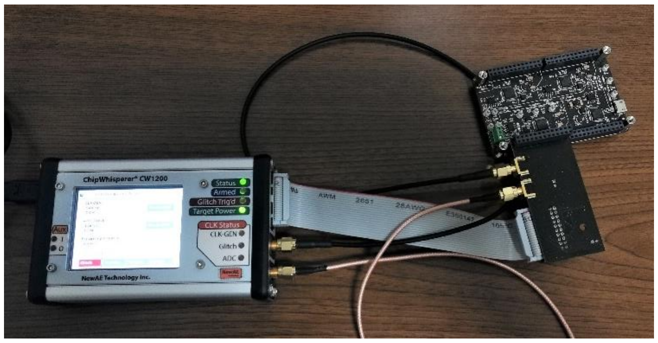


Figure 8 Picture of a RAMN board connected to ChipWhisperer Pro through a dedicated expansion board.

As the ChipWhisperer already has a rich environment for side-channel analysis, we took the time to design a PCB to easily integrate a RAMN ECU into the UFO framework provided by NewAE Technology [36]. The PCB is shown in Figure 9. Figure 10 shows an example usage of the ChipWhisperer UFO framework with a RAMN ECU. We could confirm that all ChipWhisperer functions (e.g., trigger, clocks, UART communication) would work as intended and that good quality waveforms could be obtained.

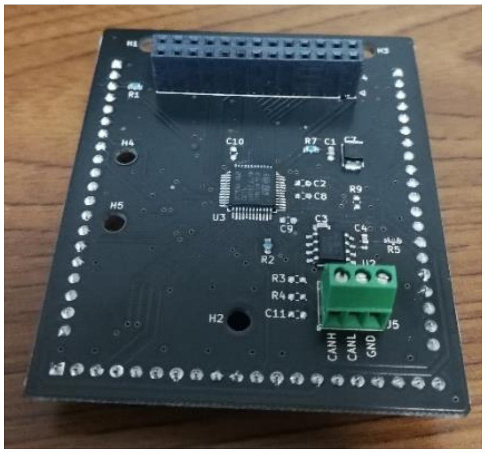


Figure 9 Picture of a RAMN ECU designed to fit into the UFO framework of ChipWhisperer.

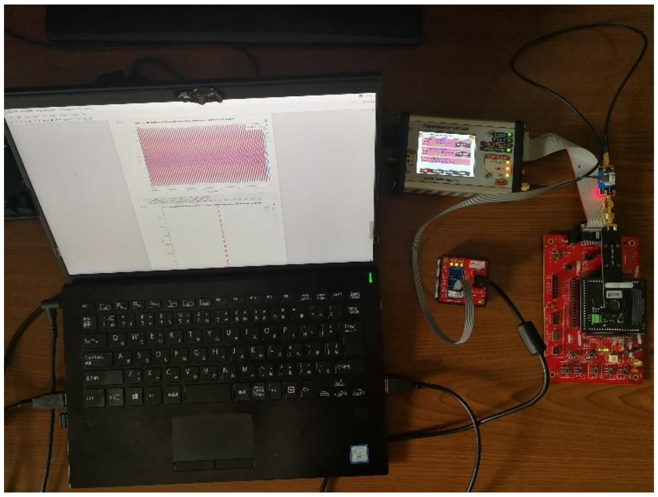


Figure 10 Picture of the RAMN UFO ECU used with ChipWhisperer Pro and its H-Field probe.

V. Limitations

In this section, we discuss the limitations of our testbed. Because we wanted to keep the board small, open source, inexpensive, and standalone, we had to make some compromises that limit its use cases.

A. No 12 V battery line

ECUs are usually powered by a 12 V battery and need to stay functional over a wide supply voltage range (e.g., 6 V to 16 V). However, supplying 16 V from a USB port would put harder requirements on the four power supplies, resulting in a bigger, noisier, and more expensive board. Therefore, we decided to directly use the 5 V USB port. This limitation prevents researchers from researching information leaks on the 12 V battery line. However, this does not impact the CAN/CAN-FD, which uses 5V by design.

B. Only one CAN/CAN-FD bus

Contrary to PASTA, which features physically separated CAN buses connected indirectly through a gateway ECU, our testbed only features one common CAN/CAN-FD bus. This means that researchers cannot easily use it for research related to gateway ECU, such as CAN/CAN-FD firewalls. However, they can address this issue by connecting several RAMN testbeds together.

C. Not 100% automotive grade.

To keep the testbed open source, we chose a publicly available microcontroller instead of an actual automotive-grade microcontroller. To limit the impact of this factor, we tried to select a microcontroller that has a wide temperature range and many safety features, thus making it "close enough" to an automotive-grade one. Ideally, the testbed would feature an AEC-Q100 qualified microcontroller, with automotive security elements, such as an EVITA hardware security module, a secure hardware extension (SHE), or an automotive TPM. However, this is not currently possible because of NDA limitations, which we hope will be lifted in the future.

VI. Conclusion

We developed and evaluated RAMN, a credit card-sized automotive security testbed similar to PASTA [27], but designed with different goals. We kept the testbed inexpensive to facilitate destructive and nonreversible testing. We used mostly automotive-grade components to ensure that the testbed has characteristics close to those of real ECUs and that it can operate in extreme conditions. RAMN can be used in conjunction with CARLA [43] to simulate an active automotive network of a self-driving vehicle. Because it is small, researchers can use the testbed in special environments, such as clean rooms, and fit it into testing equipment, such as microprobing stations. It is inexpensive, and therefore researchers do not need to share it with others and can work from home. They can also perform potentially destructive attacks without worrying about their budget, and they can build dozens of them to evaluate the effects of manufacturing tolerances. We optimized RAMN for physical testing, including side-channel analysis and glitching attacks. As a result, this testbed offers more freedom for researchers to evaluate attacks and countermeasures involving physical parameters. In the future, we would like to explore the testbed's possibility for education and bug bounty programs. To keep the board small, open source, and inexpensive, we had to make some compromises that limit the use cases of the board. Most of these compromises can be addressed in future works, for example, using a slightly bigger and more expensive testbed. However, the inaccessibility of automotive-grade microcontrollers and automotive-grade software is a bigger issue that cannot be fixed easily as of 2020. By releasing this testbed, we hope to contribute another step toward a more open automotive security industry.

VII. Bibliography

[1] GRIMM & Scythe, "What is “3PO”?," 17 07 2017. [Online]. Available: https://blog.grimm-co.com/post/what-is-3po/. [Accessed 02 10 2019].

[2] Bugcrowd, "https://www.bugcrowd.com/resources/webinars/from-an-ivi-in-a-box-to-a-car-in-a-box/," [Online]. Available: https://www.bugcrowd.com/resources/webinars/from-an-ivi-in-a-box-to-a-car-in-a-box/.

[3] S. Skorobogatov, "Hardware Security of Semiconductor Chips: Progress and Lessons," in *NCL*, 2011.

[4] ISO, "ISO 26262-1:2018 Road Vehicles — Functional Safety," [Online]. Available: https://www.iso.org/standard/68383.html. [Accessed 21 10 2019].

[5] Automotive Electronics Council, "AEC-Q100 Rev G - Failure Mechanism Based Stress Test Qualification For Integrated Circuits," 14 05 2007. [Online]. Available: http://www.aecouncil.com/Documents/AEC_Q100_Rev_G_Base_Document.pdf.

[6] Automotive Electronics Council, "AEC-Q200 REV D Stress Test Qualification for Passive Components," 01 06 2010. [Online]. Available: http://www.aecouncil.com/Documents/AEC_Q200_Rev_D_Base_Document.pdf.

[7] Renesas, "RH850 Family," [Online]. Available: https://www.renesas.com/eu/en/products/microcontrollers-microprocessors/rh850.html. [Accessed 16 10 2019].

[8] Infineon, "32-bit AURIX Microcontroller Based on TriCore," [Online]. Available: https://www.infineon.com/cms/en/product/microcontroller/32-bit-tricore-microcontroller/. [Accessed 16 10 2019].

[9] C. Turner, "Safety and Security for Automotive SoC Design," 28 06 2016. [Online]. Available: https://www.arm.com/files/pdf/20160701_B01_ATF_Taiwan_Chris_Turner.pdf.

[10] ISO, "ISO/IEC TS 15504-10 Information Technology — Process Assessment," [Online]. Available: https://www.iso.org/standard/54537.html. [Accessed 21 10 2019].

[11] MISRA, "MISRA Publications," [Online]. Available: https://www.misra.org.uk/Publications/tabid/57/Default.aspx.

[12] Green Hills, "Green Hills Platforms for Automotive," [Online]. Available: https://www.ghs.com/products/auto_solutions.html. [Accessed 16 10 2019].

[13] Wind River, "Development Tools," [Online]. Available: https://www.windriver.com/products/development-tools/. [Accessed 16 10 2019].

[14] Elektrobit, "EB Tresor Product Line," [Online]. Available: https://www.elektrobit.com/products/ecu/eb-tresos/. [Accessed 16 10 2019].

[15] ETAS, "RTA-OS," [Online]. Available: https://www.etas.com/en/products/rta_os.php. [Accessed 16 10 2019].

[16] AUTOSAR, "General Information About AUTOSAR," 16 10 2019. [Online]. Available: https://www.autosar.org/about/.

[17] Wikipedia, "OSEK," [Online]. Available: https://en.wikipedia.org/wiki/OSEK. [Accessed 21 10 2019].

[18] "ISO/TS 16949 Quality Management Systems," 2009. [Online]. Available: https://www.iso.org/standard/52844.html.

[19] R. Oshiro, "Fundamentals of AEC-Q100: What “Automotive Qualified” Really Means," 11 2018. [Online]. Available: https://media.monolithicpower.com/mps_cms_document/w/e/Webinar_-_Fundamentals_of_AEC-Q100-6Nov2018.pdf.

[20] MISRA, "MISRA C:2012 – Addendum 3 Coverage of MISRA C:2012 (Including Amendment 1) against CERT C 2016 Edition," [Online]. Available: https://www.misra.org.uk/LinkClick.aspx?fileticket=PLS6_EzbAl0%3D&tabid=57.

[21] P. Nasahl, N. Timmers, "Attacking AUTOSAR using Software and Hardware Attacks," [Online]. Available: https://pure.tugraz.at/ws/portalfiles/portal/23511745/Attacking_AUTOSAR_using_Software_and_Hardware_Attacks.pdf. [Accessed 23 10 2019].

[22] A. Milburn, N. Timmers, N. Wiersma, R. Pareja, S. Cordoba, "There Will Be Glitches: Extracting and Analyzing Automotive Firmware Efficiently," [Online]. Available: https://www.riscure.com/uploads/2018/11/Riscure_Whitepaper_Analyzing_Automotive_Firmware.pdf.

[23] L. Cojocar, K. Razavi, C. Giuffrida, H. Bos, "Exploiting Correcting Codes: On the Effectiveness of ECC Memory Against Rowhammer Attacks," in *IEEE S&P*, 2019.

[24] J. Bauer, M. Gruhn, and F. C. Freiling, "Lest We Forget: Cold-Boot Attacks on Scrambled DDR3 Memory," in *Digital Investigation, vol. 16*, 2016.

[25] J. Munera, J. M. de Fuentes, A. I. González-Tablas, "Towards a Comparable Evaluation for VANET

Protocols: NS-2 Experiments Builder Assistant and Extensible Test Bed," in *escar Europe*, 2011.

[26] K. Fischer, "High Assurance Cyber Military Systems (HACMS)," in *escar USA*, 2013.

[27] T. Toyama, T. Yoshida, H. Oguma, T. Matsumoto, "PASTA: Portable Automotive Security Testbed with Adaptability," in *Black Hat Europe*, 2018.

[28] Thomas Korak, Michael Hutter, Barıs Ege, Lejla Batina, "Clock Glitch Attacks in the Presence of Heating," in *FDTC*, 2014.

[29] "Smart Card Fault Injections with High Temperatures," in *FCTRU*, 2016.

[30] Naghmeh Karimi, Thorben Moos, Amir Moradi, "Exploring the Effect of Device Aging on Static Power Analysis Attacks," in *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019.

[31] Naghmeh Karimi, Sylvain Guilley, Jean-Luc Danger, "Impact of Aging on Template Attacks," in *GLSVLSI*, 2018.

[32] Toshihiro Katashita, Yohei Hori, Hirofumi Sakane, Akashi Satoh, "Side-Channel Attack Standard Evaluation Board SASEBO-W for Smartcard Testing," 2011.

[33] R. Velegalati and J.-P. Kaps, " Introducing FOBOS: Flexible Open-source BOard for Side-channel analysis," in *Third International Workshop on Constructive Side-Channel Analysis and Secure Design*, 2012.

[34] J. Kim, K. Oh, D. Choi, and H. Kim, "SCARF: Profile-Based Side Channel Analysis Resistant," in *Security and Management International Conference, SAM'12*, 2012.

[35] C. O'Flynn, "A Framework for Embedded Hardware Security Analysis," 2017.

[36] C. O'flynn, "CW308 UFO Target," 2016. [Online]. Available: http://wiki.newae.com/CW308_UFO_Target.

[37] L. Culhane, "Serial Line CAN Interface Driver (Using tty Line Discipline)," 08 10 2019. [Online]. Available: https://github.com/torvalds/linux/blob/master/drivers/net/can/slcan.c. [Accessed 08 10 2019].

[38] kernel.org, "Readme File for the Controller Area Network Protocol Family (aka SocketCAN)," [Online]. Available: https://www.kernel.org/doc/Documentation/networking/can.txt. [Accessed 14 09 2020].

[39] STMicroelectronics, "USB DFU Protocol Used in the STM32 Bootloader," 08 10 2019. [Online]. Available: https://www.st.com/resource/en/application_note/cd00264379.pdf. [Accessed 08 10 2019].

[40] STMicroelectronics, "STM32 Nucleo Boards," [Online]. Available: https://www.st.com/ja/evaluation-tools/stm32-nucleo-boards.html. [Accessed 14 09 2020].

[41] Amazon Web Services, Inc, "FreeRTOS Real-Time Operating System for Microcontrollers," [Online]. Available: https://www.freertos.org/. [Accessed 24 08 2020].

[42] Adafruit, "Adafruit 1.3" 240x240 Wide Angle TFT LCD Display with MicroSD - ST7789," [Online]. Available: https://www.adafruit.com/product/4313. [Accessed 14 09 2020].

[43] C. Team, "CARLA Open-Source Simulator for Autonomous Driving Research.," [Online]. Available: https://carla.org/. [Accessed 24 08 2020].

[44] Toyota Motor Corporation, "CAN ID List V1.0E," 25 3 2019. [Online]. Available: https://github.com/pasta-auto/PASTA1.0/blob/master/doc/PASTA1.0%20CAN-ID%20List%20v1.0E.pdf.